Artículo Científico / Scientific Paper

INGENIUS
Revista de Ciencia y Tecnología
DOI: 10.17163/ings.n14.2015.02

Emergency-route, evacuation route finder for emergency and rescue scenarios

Emergency-route, buscador de rutas de evacuación para escenarios de emergencia y rescate

Rommel V. Torres[1,*], Franscisco Sandoval-Noreña[2], Vicente M. Martínez[3]

Resumen

Con el creciente ritmo de desarrollo de aplicaciones para dispositivos móviles se dispone de diversos sistemas de localización. El GPS es el de mayor utilización, permite conocer la ubicación actual de un dispositivo en cualquier momento, además, posibilita el desarrollo de soluciones para una interacción más cercana entre dos o más personas independientemente del entorno geográfico en el que se encuentren. Esta investigación permitió desarrollar una aplicación móvil basada en el sistema operativo Android, con la finalidad de ayudar a los usuarios en las tareas de evacuación en situaciones de emergencia, presentando una ruta de salida óptima desde la ubicación actual del usuario hacia la zona de evacuación más cercana. Las rutas son calculadas y transmitidas a los usuarios en relación con los obstáculos reportados y las zonas seguras establecidas.

Palabras clave: aplicaciones en red para desastres, escenarios de emergencia y rescate, redes móviles.

Abstract

With the increasing pace of application development for mobile devices, several location systems are available, being GPS widely used, which allows to know the actual location of a device at any time, besides it makes it possible to develop solutions allowing a close interaction among two or more people regardless the geographical position they are. The present research allowed us to develop a mobile application based on Android OS, in order to help users evacuate at emergency situations, showing an optimal exit route from the actual location to the closest evacuation zone. Routes are calculated and transmitted to users based on reported obstacles and pre-established safety zones.

Keywords: Disaster network applications, emergency and rescue, mobile networks.

[1,*]Doctor por la Universidad Politécnica de Madrid, ingeniero en Informática, docente - Universidad Técnica Particular de Loja, Departamento de Ciencias de la Computación y Electrónica, sección de Electrónica y Telecomunicaciones. Autor para correspondencia ✉: rovitor@utpl.edu.ec.
[2]M. Sc. en Ingeniería Eléctrica, ingeniero en Electrónica y Telecomunicaciones, docente - Universidad Técnica Particular de Loja, Departamento de Ciencias de la Computación y Electrónica, sección de Electrónica y Telecomunicaciones.
[3]M. Sc. Ingeniero en Electrónica y Telecomunicaciones, docente - Universidad Técnica Particular de Loja, Departamento de Ciencias de la Computación y Electrónica, sección de Electrónica y Telecomunicaciones.

Recibido: 13-11-2015, aprobado tras revisión: 24-11-2015.

Forma sugerida de citación: Torres, R.; Sandoval-Noreña, F.; Martínez, V. (2015). "Emergency-route, evacuation route finder for emergency and rescue scenarios". Ingenius. N.°14, (Julio-Diciembre). pp. 14-20. ISSN: 1390-650X.

1. Introduction

Today, the important deployment that communication technologies have had, involves a different view about how people communicate with their social environment, in such way that distance and geographic location are not determining impairments anymore. With a continuous progress of technologies and huge user's collaboration, it is possible to have at our disposal any information that maybe it was impossible to obtain in the past. Furthermore, it is important to consider that mobility is becoming a prior characteristic, and also applications that allows users to interact each other regardless their situation (chat, calls, leisure, emergency situations, warnings, messages, etc.) maintaining two important characteristics: portability and ease [1]. Mobility is not only a matter of communication, but also a matter of emergency and rescue, where it is very important to dispose elements which help us to find the closest evacuation zone, and easing rescue operations to the personnel. The quick expansion of smartphones in the market as well as the operative system developer's community allows the opportunity to explore development kits and their offered possibilities to develop applications which can help to their users in emergency situations, specifically to evacuate to safety zones during complex situations as natural disasters or any other situation where evacuation is needed. This paper is organized in the following manner: a review about mobile applications applied to emergency situations is shown in Section II, a proposed solution is shown in Section III and later its development is shown in Section IV. Then, testing of application performance is shown in Section V, and finally conclusions are shown.

2. Review about Mobile Applications in Emergency Situations

Nowadays, technology consumers have a wide assortment of devices, which operate with different operating systems: Android (from Google), iOS (from Apple), Symbian OS (from Nokia), Blackberry OS (from RIM) and Windows Phone (from Microsoft). After a market share analysis, iOS from Apple keeps the first place, including all its devices (iPhone, iPad, iPod) [2]. On the other hand, Android keeps the first place considering smartphones sales only [3]. Now, if we consider the number of available applications, iOS exceeds Android. However, Android offers a wide range of free applications and it keeps growing quickly, being almost one million of regular application in May 2014 [4]. In [5] a mobile application review is approached (Android), for dealing with emergency situations. Among others, it is remarked that, from reviewed applications in [5], 50.8 % has a functional objective to ask for help when users suffer a health issue, (for instance, heart attack, epilepsy, hypertension, diabetes), 23.2 % asks for help to police force or security personnel, whereas 3.6 % gives users a set of instructions about how to evacuate from an emergency location. Among existing applications oriented to emergency assistance (we consider the most relevant applications to our opinion, in relationship to the application that we propose to develop, and considering several operating systems) we can mention:

iRescue: Gives assistance to rescue teams obtaining information about people evacuation in traffic accidents. It allows to visualize the position of the device on a map and sends an image containing the GPS position of the device by email.

Emergency Alarm: This is a useful tool for alarming urgent events via SMS, even when the device is in silent mode. It notifies a relative or friend about an issue, and also it can be used in disasters like earthquakes, floods, or security threats for notification purposes.

Rescue Kit: it obtains emergency contacts of almost every country in the world, depending on the actual location, and users can use them. We can remark some features:

It allows visualization about actual locations and instant notification to the closest rescue agency.

It shares the location by using SMS, email or social network.

All the applications presented above required a continuous Internet connection, because the information is shared through the Internet, in order to send or receive data. Our solution has a different option which does not need an Internet connection for operation, considering that in many cases the occurrence of any issue can involve a temporal disconnection from the network like cellular network of data network.

3. Proposed solution

Until today, most of the available solutions for emergency evacuation shared data based on web services or APIs (Application Programming Interface) remotely. They restrict their use to environments where there is access to a data network only. This situation can differ depending on the emergency environment or disasters, where there can be a loss of communication services.

The proposed work is focused in the development of an application that suits to environments where there is no internet access. In contrast, there is the need of a local wireless network, from which the user can visualize the closer evacuation locations, avoiding obstacles reported by other users, which do not allow the regular personal or car displacement. Also, it allows to rescue equipment to find victims with reduced mobility.

For designing the application's operational environment, we will consider the following points:

Coverage: Application's coverage depends on the wireless network predefined.

Visibility: while using a wireless data network, there is no need of having a line of sight between devices.

Number of nodes: the number of nodes within the network depends on the number of connections supported by the wireless network, since the application does not keep a continuous connection with every node, but it works on demand.

In Fig. 1 the proposed solution model is shown, which includes four principal elements: users, application, data available to the user within the application, and the wireless network.

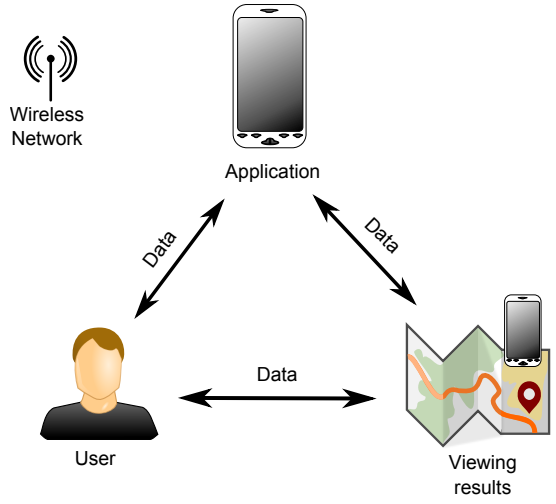


Figure 1. Diagram of proposed solution.

Users: We consider as user to each person within the coverage area that is using the application.

Application: Is the solution to develop, that will be installed in mobile devices working on Android, and it may work under a local wireless network without Internet access.

Results view: It represents to all data available to the user on the mobile device.

Wireless network: Implemented by a user's device or external equipment.

The information about possible existing obstacles can be found on server node and it contains all the information available at that moment. Besides, this information is available to all users, thus they can check it at any time and they can report the existence of new obstacles or secure zones. By using the information obtained by the application, there is possible to draw the optimal exit route from the actual position. Each time a new obstacle or secure zone is reported, the algorithm for searching the best exit route is redefined. Each user can search the existence of new obstacles, users, map or exit route at any time. Data synchronization can be performed manually or automatically.

The proposed solution involves three levels: a presentation layer, which is the graphic interface where the user can visualize the provided information and enter new data, a logic business layer, which define the exchange of information between model and presentation layer, and data access layer.

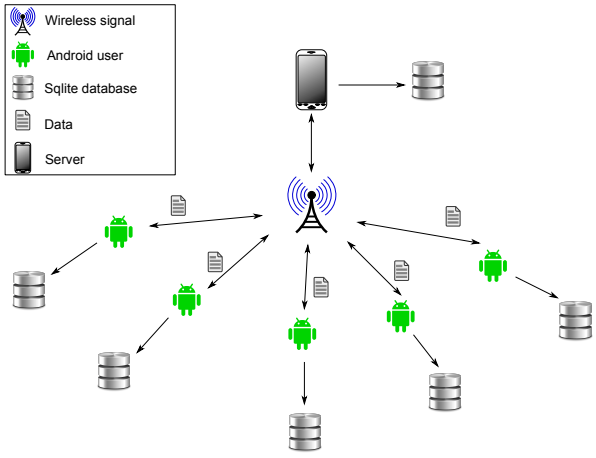


Figure 2. Application Topology.

In Fig. 2 is shown the proposed functional topology, where user's interaction can be visualized and the data stored in the local database. The proposed wireless signal is external, which means that it can be accessible by any other existing computer equipment. The server proposed in the architecture is a common mobile device which goal is to save the information sent to the rest of the nodes. The information that this server has is about the existing users in the network, reported obstacles and zones reported as secured zones. Finally, the database allows storing the received information from a mobile device (server), and there is one by each user. Therefore, for the proper operation of the application, it is required that the mobile device with the information to synchronize is working in the wireless network, and the users must access to the device mention above for downloading map data, existing obstacles and network users.

4. Solution deveplopment

In order to develop the application, we used the following tools:

- Eclipse ID 3.7 [6],
- SDK tools Android [7],
- Android S.O. v4.1 [8],
- Mobac (Mobile Atlas Creator) Osmdroid android [9],
- GPS receiving antenna,
- Gpx files,
- Sqlite database [10].

Eclipse ID 3.7 Indigo, is a multi-platform open source Integrated Development Environment (IDE) for business application development under several programming languages such as Java, PHP, Python, C, C++, Ruby, Javascript, etc. SDK tools Android provides API libraries (Application Programming Interface) and enough development tools to build, test and debug applications for Android devices. Furthermore, Android OS v4.1, is a Linux-based operating system, design mainly for touch screen devices as smartphones and tablets. Android OS structure is composed of applications which are executed in a Java application framework object-oriented over a Java library core in a Dalvik virtual machine with real-time compilation [11]. Mobac (Mobile Atlas Creator), is a multi-platform open source application which creates maps for GPS devices and mobile applications like TrekBuddy (http://www.trekbuddy.net/), AndNav (http://www.andnav.org) and other Android and WindowsCE applications to use them without an internet connection. Osmdroid-android is a library developed in Java which allows the use of the OpenStreetMap API for rendering the map in desktop applications or via web. The GPS receiver antenna is an essential component that has to include the mobile device that is going to use the application. It is used to obtain the actual location and send it to the other users when an obstacle is reported at any time. GPS system is composed of three segments: spatial, control Ald user [12]. .Gpx or GPS eXchange Format device is an XML scheme for GPS data transferring between applications. It can be used to describe waypoints, tracks, and routes. Finally, SQLite is a processing library which implements an autonomous SQL database, without server, configuration or transactional.

4.1. Obtaining the Evacuation Route

The evacuation route considers user location, evacuation point location, obstacles, and possible existing routes. Based on this information, it calculates point-to-point distances through an algorithm, and determines the most efficient route. These problems are solved in the following manner.

4.1.1. Point-to-point distance calculation

In order to calculate the distance between two points, two geographic coordinates are needed, the starting point (user location) and the end point (evacuation location). Both points must have predefined their geolocation coordinates. Considering the earth curvature, Haversine formula is needed [13], which is

$$\text{haversine}\left(\frac{d}{r}\right) = \text{haversine}\left(\phi_2 - \phi_1\right) + \ldots$$

$$\ldots + \cos(\phi_1) \cdot \cos(\phi_1) \cdot \text{haversine}(\lambda_2 - \lambda_1) \qquad (1)$$

where havsin represents the haversine function, represented by

$$\text{haversine} = \sin^2\left(\frac{\theta}{2}\right) = \frac{1 - \cos\theta}{2} \qquad (2)$$

Where d is the distance between two points (spherical distance), r is the earth radius, which is 6378 km, assuming that the earth is perfectly round (equatorial radius is 6378.14 km and polar radius is 6356.78 km). ϕ_1 and ϕ_2 are initial point latitude (user) and end point (evacuation location) respectively, and λ_1 and λ_1 are the initial point and end point longitude respectively. On the left side of the equation 1, d/r term is the central angle, assuming that it is measured in radians (an early conversion of latitudes and longitudes to radians is needed, which are expressed in degrees, minutes and seconds) is possible to solve the equation 1 for d, applying the inverse haversine function or inverse sine function:

$$d = r \cdot \text{haversin}^{-1}(h) = 2\,r \cdot \arcsin\left(\sqrt{h}\right) \qquad (3)$$

where $h = \text{haversin}^{-1}(d/r)$ and it must be a number between 0 and 1 for d to be real, or explicitly

$$d = 2r \cdot \arcsin\left[\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \ldots\right.$$

$$\left.\ldots + \cos(\phi_1) \cdot \cos(\phi_2) \cdot \sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)\right]^{\frac{1}{2}} \qquad (4)$$

The implementation of the distance calculation between starting point and ending point is shown in Algorithm 1.

4.1.2. Exit Routes Matrix

In order to find routes where users can get the exit from the actual location, we use Mobac tool which generates a matrix that includes the main existing points inside an area for calculating the routes. This matrix has several points located at street intersections within a predefined area. Then, the best route is calculated through Dijkstra algorithm, which is explained below.

4.1.3. Exit routes search

Dijkstra algorithm is an efficient algorithm (of complexity $O(n^2)$ where n is the number of vertices) which is used to find the minimum-cost route from an initial node to the rest of the graph nodes [14]. It was designed by the Dutch Edsger Wybe Dijkstra in 1959. Algorithm 2 shows the Dijkstra algorithm implementation, which works in a recursive manner and it is responsible for finding and tracing the closest evacuation route from the initial point to the closest evacuation zone. At each iteration, the algorithm chooses the best option from the available ones in order to find the best global solution.

Algorithm 1. Distance calculation between two coordinates.

```
/**
@param lon1. Origin point length
@param lat1. Origin point latitude
@param lon2. Destination point length
@param lat2. Destination point
    latitude
@return. Distance between two points,
    origin and destination.
*/
private static double
CalcularDistanciaHaversine(double
    lon1, double lat1,
double lon2, double lat2) {
double radio = 6378; // km Radio de
    la Tierra
lat1 = Math.toRadians(lat1);
lon1 = Math.toRadians(lon1);
lat2 = Math.toRadians(lat2);
lon2 = Math.toRadians(lon2);
double dlon = (lon2 - lon1);
double dlat = (lat2 - lat1);
double sinlat = Math.sin(dlat / 2);
double sinlon = Math.sin(dlon / 2);
double a = (sinlat * sinlat) +
    Math.cos(lat1)*Math.cos(lat2)*
(sinlon*sinlon);
double c = 2 * Math.asin
    (Math.min(1.0, Math.sqrt(a)));
double distancia = radio * c * 1000;
return distancia;
}
```

Algorithm 2. Dijkstra algorithm implementation.

```
/**
@param inicio. Inicial node.
@param fin. Destination node.
*/
publicString
    encontrarRutaMinimaDijkstra(String
    inicio, String fin)
{
// calcula la ruta mas corta del
    inicio a los demas
encontrarRutaMinimaDijkstra(inicio);
// recupera el nodo final de la lista
    de terminados
Nodo tmp = new Nodo(fin);
if(!listos.contains(tmp))
{
System.out.println("Nodo no
    alcanzable"); return "";
}
tmp = listos.get(listos.indexOf(tmp));
Double distancia = tmp.distancia;
// crea una pila para almacenar la
    ruta desde el nodo
//final al origen
Stack<Nodo> pila = new Stack<Nodo>();
while(tmp != null)
{
pila.add(tmp);
tmp = tmp.procedencia;
}
String ruta = "";
// recorre la pila para armar la ruta
    en el orden correcto
while(!pila.isEmpty())
ruta += (pila.pop().id + ";");
return ruta;
}
```

5. Operational Tests

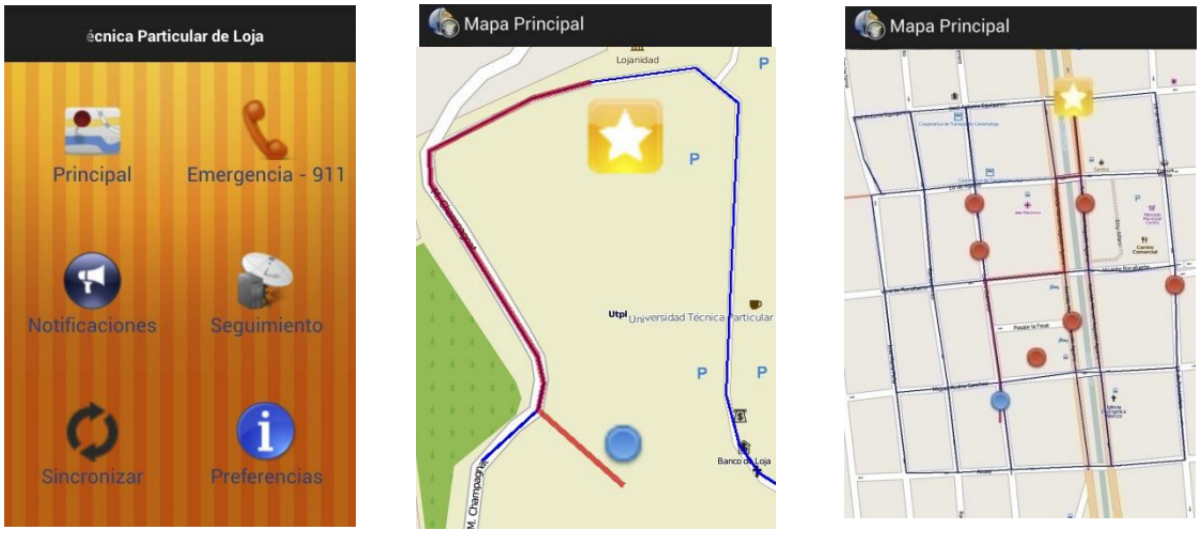
In order to perform tests, we chose a predefined area, we generated an offline map by using Mobac, and we use devices with Android 4.0.1 O.S. On each device, we installed the application to check its operation and further evaluation following the requirements below:

1. One node must be set up as Master, which means that the rest of devices will connect to it as a first step, working as an access point.
2. All nodes must have access to the available wireless network.
3. The application must be correctly installed on all devices.
4. The device must have an activated GPS system which should work properly.
5. The application must download the map previously generated by Mobac at each device.

5.1. Operating Scenarios

We define two operating scenarios in order to test the application. The first scenario is shown in Fig. 3b where an urban area has been selected in downtown Loja (located in southern Ecuador, South America), in which the street distribution is better defined than rural area. This area has better features in terms of route visibility and location.

The second scenario is shown in Fig. 3c, which UTPL University Campus has been selected. This campus is located in San Cayetano Alto, a neighborhood in Loja city. In this operating scenario, the main goal is to visualize the exit route trace in emergency case, where there are no predefined routes.



(a) Main menu **(b)** Scenario 1: Downtown Loja **(c)** Scenario 2: UTPL Campus

Figure 3. Data visualization.

5.2. Application Interface

When the application starts, the user can choose among several available menus, shown in Fig. 3a, which are described below:

- Principal (main): objects that will appear in the map.
- Emergencia (emergency): 911 emergency call.
- Notificaciones (notifications): sent and received notifications.
- Seguimiento (monitoring): route chosen by the user and already registered.
- Sincronización (sync): sync options with other devices.
- Preferencias (preferences): user preferences.

5.3. Object view on the map

In Fig. 3b we can see several obstacles (red dots), the user's actual location (blue dot), the closest evacuation zone (white start within a yellow square) and an exit route calculated by the application, which user must follow in order to arrive to the evacuation zone (red line). Each object on the screen will change or update while the user is walking or moving.

5.4. Types of Data Synchronization

There are several types of data synchronization, depending on the object that we want to synchronize: maps, obstacles, users and evacuation points. Each sync event executes depending on the user's decision (manual synchronization) or each amount of time selected by the user (automatic synchronization).

6. Conclusions

This research shows the development of an emergency and rescue application for mobile devices, which goal is to reduce evacuation time of people to secure zones in disasters. There is no need of data connection; the application needs a local wireless network which allows a synchronization service between devices. Android operating system has been used and geolocation features from mobile devices have been considered in order to define user's location. Operating tests showed positive results in terms of functionality and user interaction with the application, offering the opportunity to report or remove obstacles, evacuation points to the user, and to obtain a secure evacuation route without the need of data connection. This application is the first in its category because it uses a communication strategy based on device communication capabilities without depending on a predefined communication structure.

Acknowledgement

We wish to acknowledge the contribution of Luis Sancho in completing this research. This article has also been supported in part by Senescyt-Ecuador. An earlier version of this article was presented in Extremecom 2014 at the national researchers track.

References

[1] R. Torres, L. Mengual, O. Marban, S. Eibe, E. Menasalvas, and B. Maza, "A management Ad Hoc networks model for rescue and emergency scenarios," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9554–9563, 2012.

[2] A. Asthana and R. Asthana. IOS 5, Android 4.0 and Windows 8 - A Review. BEACON IEEE 31, 33-43, 2012.

[3] Gartner's report. (2012) Market share: Mobile devices, worldwide. [Online]. Available: http://www.gartner.com/resId=2117915.

[4] Android market stats. Number of available apps. [Online]. Available: http://www.appbrain.com/stats/.

[5] D. Gómez, A. Bernardos, J. Portillo, P. Tarrío, and J. Casar, *Highlights on Practical Applications of Agents and Multi-Agent Systems.* Springer Berlin Heidelberg, 2013, ch. A Review on Mobile Applications for Citizen Emergency Management, pp. 190–201.

[6] Eclipse Foundation. (Mayo 2014) Eclipse indigo. [Online]. Available: http://www.eclipse.org/indigo/

[7] Android. (Mayo 2014) SDK Tools. [Online]. Available: http://developer.android.com/tools/sdk/tools-notes.html

[8] ——. (Mayo 2014) Android 4.1 APIs. [Online]. Available: http://developer.android.com/about/versions/android-4.1.html

[9] Mobac. (Mayo 2014) Mobile atlas creator. [Online]. Available: http://mobac.sourceforge.net/

[10] SQlite. (Mayo 2014) SQlite. [Online]. Available: http://www.sqlite.org/

[11] Google, Inc. (Mayo 2014) What is android? [Online]. Available: https://developer.android.com/.about/index.html

[12] US Government. (Mayo 2014) Global positioning system. [Online]. Available: http://www.gps.gov/

[13] C. Veness. (January 2012) Calculate distance and bearing between two latitude/-longitude points using haversine formula in Javascript. [Online]. Available: http://www.movable-type.co.uk/scripts/latlong.html

[14] G. Torrubia and V. Terrazas, "Algoritmo de Dijkstra. Un tutorial interactivo," in *VII Jornadas de Enseñanza Universitaria de la Informática (JENUI 2001)*, 2012.